

### Solution 1: L0 Regularization

Consider the regression learning setting, i.e.,  $\mathcal{Y} = \mathbb{R}$ , and feature space  $\mathcal{X} = \mathbb{R}^p$ . Let the hypothesis space be the linear models:

$$\mathcal{H} = \{f(\mathbf{x}) = \boldsymbol{\theta}^\top \mathbf{x} \mid \boldsymbol{\theta} \in \mathbb{R}^p\}.$$

Suppose your loss function of interest is the L2 loss  $L(y, f(\mathbf{x})) = \frac{1}{2}(y - f(\mathbf{x}))^2$ . Consider the  $L_0$ -regularized empirical risk of a model  $f(\mathbf{x} \mid \boldsymbol{\theta})$ :

$$\mathcal{R}_{\text{reg}}(\boldsymbol{\theta}) = \mathcal{R}_{\text{emp}}(\boldsymbol{\theta}) + \lambda \|\boldsymbol{\theta}\|_0 = \frac{1}{2} \sum_{i=1}^n \left(y^{(i)} - \boldsymbol{\theta}^\top \mathbf{x}^{(i)}\right)^2 + \lambda \sum_{i=1}^p \mathbb{1}_{|\theta_i| \neq 0}.$$

Assume that  $\mathbf{X}^T \mathbf{X} = \mathbf{I}$ , which holds if  $\mathbf{X}$  has orthonormal columns. Show that the minimizer  $\hat{\boldsymbol{\theta}}_{L_0} = (\hat{\theta}_{L_0,1}, \dots, \hat{\theta}_{L_0,p})^\top$  is given by

$$\hat{\theta}_{L_0,i} = \hat{\theta}_i \mathbb{1}_{|\hat{\theta}_i| > \sqrt{2\lambda}}, \quad i = 1, \dots, p,$$

where  $\hat{\boldsymbol{\theta}} = (\hat{\theta}_1, \dots, \hat{\theta}_p)^\top = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}$  is the minimizer of the unregularized empirical risk (w.r.t. the L2 loss).

For this purpose, use the following steps:

(i) Derive that

$$\arg \min_{\boldsymbol{\theta}} \mathcal{R}_{\text{reg}}(\boldsymbol{\theta}) = \arg \min_{\boldsymbol{\theta}} \sum_{i=1}^p -\hat{\theta}_i \theta_i + \frac{\theta_i^2}{2} + \lambda \mathbb{1}_{|\theta_i| \neq 0}.$$

(ii) Note that the minimization problem on the right-hand side of (i) can be written as  $\sum_{i=1}^p g_i(\theta_i)$ , where

$$g_i(\theta) = -\hat{\theta}_i \theta + \frac{\theta^2}{2} + \lambda \mathbb{1}_{|\theta| \neq 0}.$$

What is the advantage of this representation if we seek to find the  $\boldsymbol{\theta}$  with entries  $\theta_1, \dots, \theta_p$  minimizing  $\mathcal{R}_{\text{reg}}(\boldsymbol{\theta})$ ?

(iii) Consider first the case that  $|\hat{\theta}_i| > \sqrt{2\lambda}$  and infer that for the minimizer  $\theta_i^*$  of  $g_i$  it must hold that  $\theta_i^* = \hat{\theta}_i$ .

*Hint:* Show that  $g_i(\hat{\theta}_i) < 0 = g_i(0)$  and argue that the minimizer must have the same sign as  $\hat{\theta}_i$ .

(iv) Derive that  $\theta_i^* = \hat{\theta}_i \mathbb{1}_{|\hat{\theta}_i| > \sqrt{2\lambda}}$ , by using (iii) (and also still considering the case  $|\hat{\theta}_i| > \sqrt{2\lambda}$ ).

(v) Consider the complementary case of (iii) and (iv), i.e.,  $|\hat{\theta}_i| \leq \sqrt{2\lambda}$ , and infer that for the minimizer  $\theta_i^*$  of  $g_i$  it must hold that  $\theta_i^* = 0$ .

*Hint:* What is  $g_i(0)$ ? Consider  $\tilde{g}_i(\theta) = -\hat{\theta}_i \theta + \frac{\theta^2}{2} + \lambda$  which is the smooth extension of  $g_i$ . What is the relationship between the minimizer of  $g_i$  and the minimizer of  $\tilde{g}_i$ ?

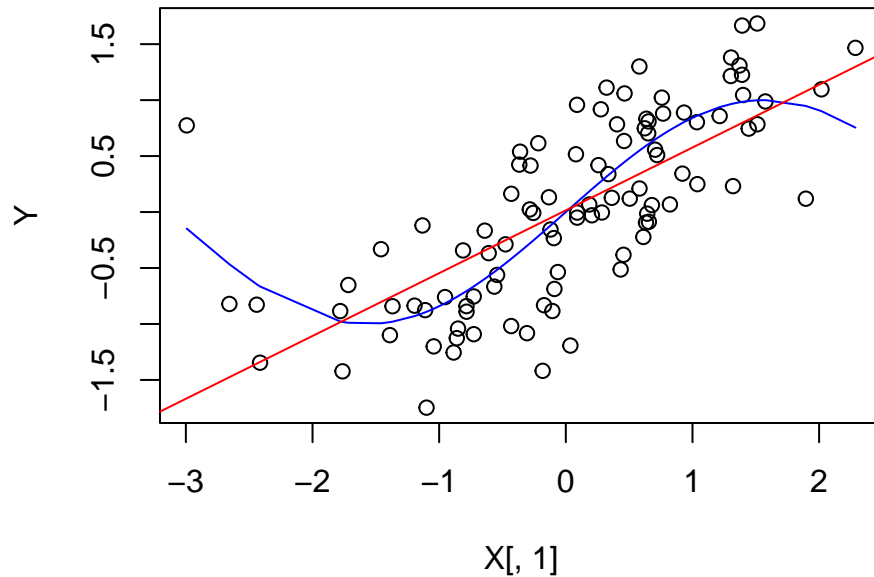
### Solution 2: Regularization

```
(a) set.seed(42)
n = 100
p_add = 100
# create matrix of features
X = matrix(rnorm(n * (p_add + 1)), ncol = p_add + 1)
Y = sin(X[,1]) + rnorm(n, sd = 0.5)
```

(b) Demonstration of

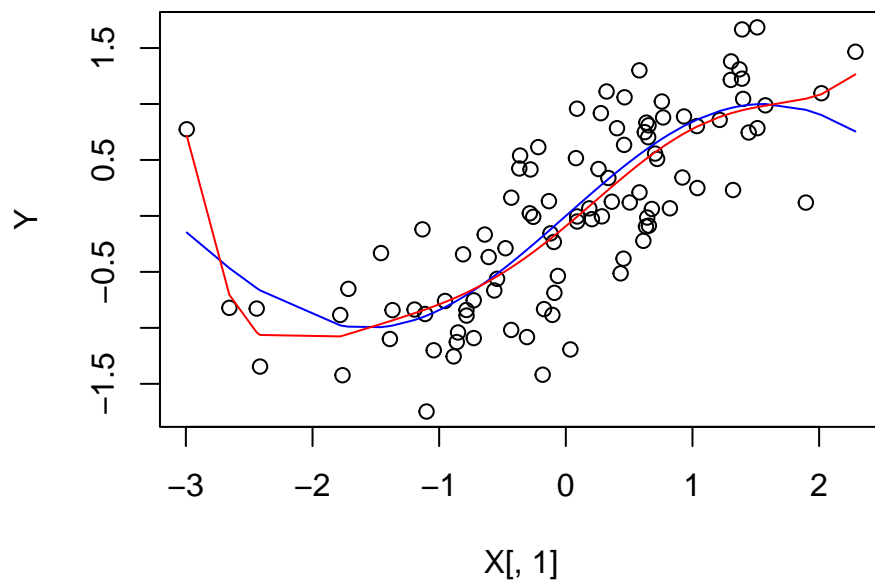
- underfitting:

```
plot(X[,1], Y)
points(sort(X[,1]), sin(sort(X[,1])), type="l", col="blue")
abline(coef(lm(Y ~ X[,1])), col="red")
```



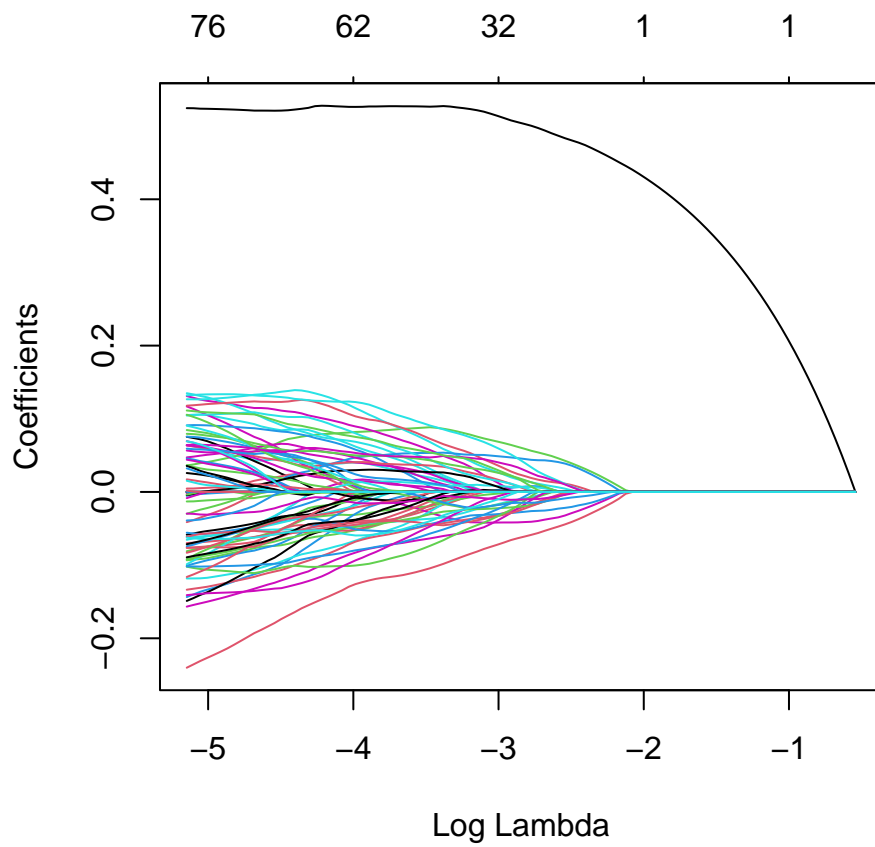
- overfitting:

```
plot(X[,1], Y)
sX1 <- sort(X[,1])
points(sX1, sin(sX1), type="l", col="blue")
points(sX1, fitted(lm(Y ~ X[,1] + I(X[,1]^2) + I(X[,1]^3) +
                      I(X[,1]^4) + I(X[,1]^5) + I(X[,1]^6) +
                      I(X[,1]^7)))[order(X[,1])],
       type="l", col="red")
```



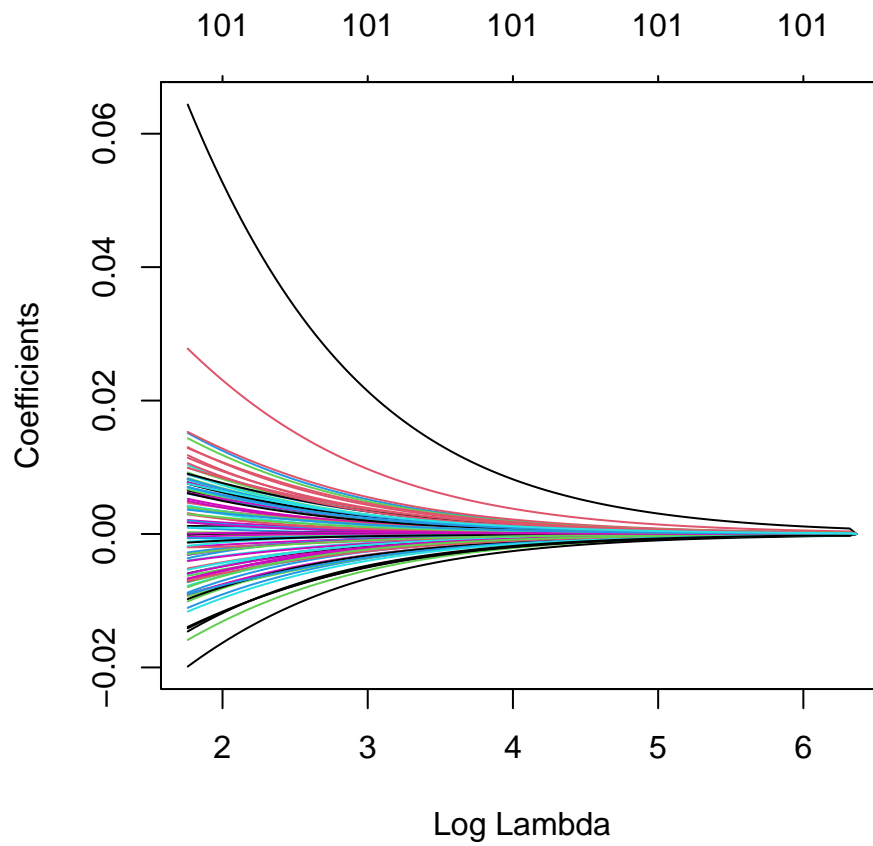
- $L1$  penalty:

```
library(glmnet)
plot(glmnet(X, Y), xvar = "lambda")
```



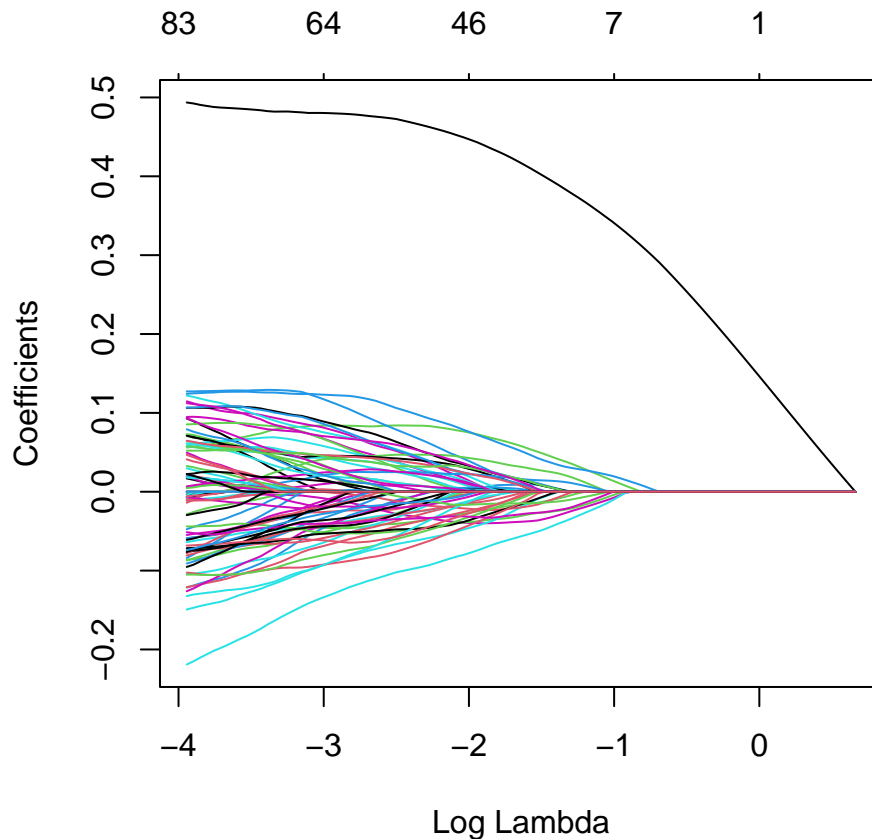
- $L_2$  penalty

```
plot(glmnet(X, Y, alpha = 0), xvar = "lambda")
```



- elastic net regularization:

```
plot(glmnet(X, Y, alpha = 0.3), xvar = "lambda")
```



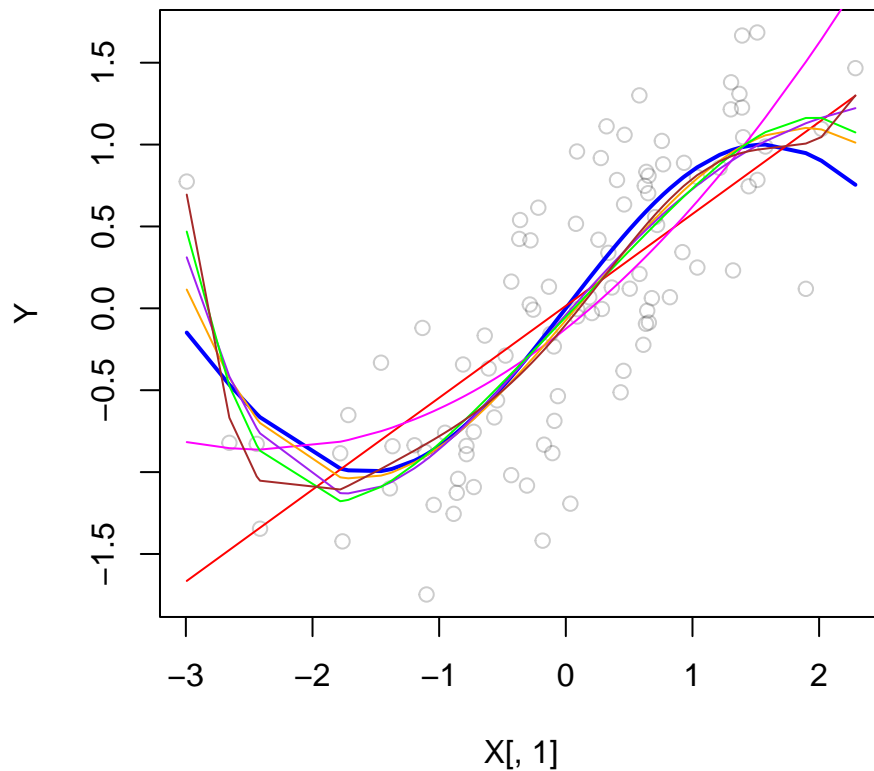
- the underdetermined problem:

```
try(ls_estimator <- solve(crossprod(X), crossprod(X,Y)))

## Error in solve.default(crossprod(X), crossprod(X, Y)) :
## system is computationally singular: reciprocal condition number = 8.30567e-18
```

- the bias-variance trade-off:

```
plot(X[,1], Y, col=rgb(0,0,0,0.2))
sX1 <- sort(X[,1])
points(sX1, sin(sX1), type="l", col="blue", lwd=2)
points(sX1, fitted(lm(Y ~ X[,1]))[order(X[,1])],
       type="l", col="red")
points(sX1, fitted(lm(Y ~ X[,1] + I(X[,1]^2)))[order(X[,1])],
       type="l", col="magenta")
points(sX1, fitted(lm(Y ~ X[,1] + I(X[,1]^2) + I(X[,1]^3)))[order(X[,1])],
       type="l", col="orange")
points(sX1, fitted(lm(Y ~ X[,1] + I(X[,1]^2) + I(X[,1]^3) +
                     I(X[,1]^4)))[order(X[,1])],
       type="l", col="purple")
points(sX1, fitted(lm(Y ~ X[,1] + I(X[,1]^2) + I(X[,1]^3) +
                     I(X[,1]^4) + I(X[,1]^5)))[order(X[,1])],
       type="l", col="green")
points(sX1, fitted(lm(Y ~ X[,1] + I(X[,1]^2) + I(X[,1]^3) +
                     I(X[,1]^4) + I(X[,1]^5) + I(X[,1]^6)))[order(X[,1])],
       type="l", col="brown")
```



- early stopping using a simple neural network:

```
library(dplyr)
library(keras)

neural_network <- keras_model_sequential()

neural_network %>%
  layer_dense(units = 50, activation = "relu") %>%
  layer_dense(units = 50, activation = "relu") %>%
  layer_dense(units = 1, activation = "relu") %>%
  compile(
    optimizer = "adam",
    loss      = "mse",
    metric    = "mse"
  )

history_minibatches <- fit(
  object      = neural_network,
  x           = X,
  y           = Y,
  batch_size  = 24,
  epochs      = 100,
  validation_split = 0.2,
  callbacks   = list(callback_early_stopping(patience = 50)),
  verbose     = FALSE, # set this to TRUE to get console output
  view_metrics = FALSE # set this to TRUE to get a dynamic graphic output in RStudio
)
```

```
## Error in eval(expr, envir, enclos): tensorflow.python.framework.errors_impl.NotFoundError:
Graph execution error:
## <... omitted ...>/site-packages/keras/optimizers/optimizer_experimental/optimizer.py",
line 1166, in _internal_apply_gradients
##     return tf.__internal__.distribute.interim.maybe_merge_call(
##     File "/Users/pmassaro/Library/r-miniconda-arm64/envs/r-reticulate/lib/python3.8/site-pack
line 1216, in _distributed_apply_gradients_fn
##     distribution.extended.update(
##     File "/Users/pmassaro/Library/r-miniconda-arm64/envs/r-reticulate/lib/python3.8/site-pack
line 1211, in apply_grad_to_update_var
##     return self._update_step_xla(grad, var, id(self._var_key(var)))
## Node: 'StatefulPartitionedCall_4'
## could not find registered platform with id: 0x110f9a900
## [[[{node StatefulPartitionedCall_4}]] [Op:__inference_train_function_927]
## See 'reticulate::py_last_error()' for details

plot(history_minibatches)

## Error in plot(history_minibatches): object 'history_minibatches' not found
```